AMENDMENT TO THE SPECIFICATION:

1. In the summary, please replace the second and third paragraphs with the following:

--In general, in one aspect, the invention features a computer-implemented methods for restructuring a design model generated from a collection of model components. The method includes receiving at a computer a command to restructure the design model by changing a hierarchical relationship of a first subset of model components with respect to the other components. In accordance with the command to restructure, a new hierarchical data structure (with corresponding new hierarchical relationships between components) is generated. Other relationships that are changed as a result of the command to restructure also are determined and these other relationships are updated and preserved subsequent to the generation of the new hierarchical data structure. The invention can be software-implemented using a computer system.--

--Implementations may include one or more of the following features. The hierarchical data structures can be formed from parent-child relationships linking a common root component with another model component. Generating the new hierarchical data structure can include changing a hierarchical path between the root component and a first subset of components. The other relationships can be, e.g., a mate relationship or an update relationship between a component in the first subset and a component outside of the first subset. Generating the new hierarchical data structure can include generating a component list that identifies a component moving to a new location. Updating the other relationships may include generating a reference list identifying the other relationships that are changed. Generating a reference list also can include associating a location code with each relationship identified ~~on~~ in the reference list. Location codes can identify a change to an associated relationship to preserve the design intent expressed by that relationship. The first subset may be, e.g., a subassembly of the model.--

2. Please replace the first two full paragraphs on page 2 with the following:

--Referring to Fig. 2, a display on the CRT 132 is shown in detail and includes a window **232**~~240~~. The window **232**~~240~~ is a conventional screen display window that can be programmed by one of ordinary skill in the art using conventional, commercially available, software programming tools, such as those available from Microsoft Corporation of Redmond, Washington.--

--The window **232**~~240~~ includes a modeling portion 242. Implementations also may include other window areas, such as a feature manager design tree 244. The modeling portion 242 contains a 3D model 257 that can be constructed and modified by the user in a conventional manner. The 3D model 257 can be displayed using solid lines and dashed lines to show visible edges and hidden edges, respectively, of the 3D model. The feature manager design tree 244 aids visualization and manipulation of the model 257 shown in the model portion 242. --

3. Please replace the first full paragraph on page 8 with the following:

--The modeled objects shown in Figs. 2 and 4 are an example of a designer's "design intent." In this case, the "intent" of the designer is to model a pair of plates held together by a bolt passing through a hole in the plates. More particularly, Fig. 2 illustrates an assembly 257 that includes an upper plate 250 and a lower plate 251, with a hole pattern (including, among others, hole 255), a bolt **258**~~253~~, and a nut 254. A "design intent" associated with the assembly 257 is to align the hole pattern on the upper plate 250 and the lower plate 251, place the bolt **258**~~253~~ through the hole 255, and attach the nut 254. The bolt **258**~~253~~ should be long enough to extend beyond the lower plate 251. This is accomplished by extending the bolt **258**~~253~~ to a plane 256 positioned a given distance beyond the bottom of the lower plate 251. Referring to Fig. 4, to express this "intent," a designer may create a subassembly **260**~~258~~that contains the bolt **258**~~253~~

and the nut 254. A positional relationship between the bolt **258~~253~~** and the nut 254 is established by creating a "mate" reference. Additionally, a size relationship (i.e., extending the bolt to the plane 256) is established by creating an update reference. Implementations can support reuse of the bolt **258~~253~~** and the nut 254 **configuration ~~conFiguration~~** by treating the subassembly **260~~258~~**as a reusable structure that may be referenced by other components. --

4. Please replace the paragraph beginning at the bottom of page 8 and continuing at the top of page9 with the following:

--An update reference exists as an external reference, meaning that the reference exists within the data structure of one component and refers to another component outside the first component. This referencing function may be implemented using, for example, a pointer structure. An update reference expresses a geometric dependency that allows one component to create (i.e., determine or set a value for), a feature in another part. One example of a feature is a dimension of a part, such as **a** lid for a container. An update reference between the container and lid can be used to ensure that changes in the container size results in an increase in the radius of the container's opening and is propagated between model components such that the radius of the lid is automatically resized.--

5. Please replace the last paragraph on page9 with the following:

--During a restructuring process, existing references between a modeled object's assemblies, subassemblies, and other components may be changed due to the rearrangement of a model's component hierarchy. To help maintain a design of a modeled object, the system **130** **~~132~~** can facilitate the maintenance of references between the object's components. This reference maintenance can include dynamically and interactively gathering and manipulating a collection

B 5

of components that form an assembly, as well as the references within the assembly. As further discussed below, design intent can be maintained during this automatic reference maintenance by determining whether relationships of components affected by restructuring operations need to be modified. If so, appropriate modifications can be automatically made.--

6. Please replace the paragraph beginning at the bottom of page 11 and continuing at the top of page12 with the following:

--Fig. 7 shows additional details for a process to construct a mate reference list 504. The process 700 locates all mate references affected by a restructuring operation and constructs the mate reference list that identifies each affected reference. The method 700 includes a search operation that may be performed for each of the source components (step 701). For each source component, the search operation 700 includes walking the modeled object's structure from the source component to the modeled object's root while examining each of the component's ancestors to locate mate references to that source component (steps 702)**, and if an ancestor has an associated mate reference, adds data indicating such association to the mate reference list**. For each source component, the walking process continues until the root is reached (steps 704-705). When the method 700 reaches the root node, the next component identifier is read from the component list and an ancestor search is performed for that source component (step 706-702). When the entire source component list has been processed, and all ancestors of these components have been examined, the method 700 returns control to the process 500 (step 707).--

B 6

7. Please replace the first full paragraph on page 12 with the following:

--The process 504 also includes constructing a list of identifiers for update references. Fig. 8 shows a process 800 that can be used to construct a list of identifiers for the update references. The process 800 includes walking (e.g., using a depth-first or breadth-first search) the entire modeled objects component hierarchy starting at the root of the assembly (step 801). Each

B 7

B7

component in the hierarchy is examined to determine whether the component contains an update reference (steps **802**-803). If a component has an update reference, an entry is made in the update reference list identifying both the referencing and the referenced components (steps 804-**805**).--

8. Please replace the paragraph beginning at the bottom of page 14 and continuing at the top of page15 with the following:

B8

SC1

--As an example, consider an update reference that establishes a dimension in one component, such as the length of a bolt, dependent upon the position of a face in a plane. The update reference may store the location of the bolt in the hierarchical data structure, store the location of the plane in the hierarchical data structure, and contain a means to specify that the geometric type referred to in the externally referenced component is a face. Modification to the update reference data **(Fig. 5, step 506)** ***(also the last sentence in the previous paragraph)*** depends on whether (i) the component containing the reference (i.e., storing the reference as an attribute) is a source component or, (ii) the component containing the reference is a non-source component that refers to a source component. Maintenance of update reference data can distinguish these cases. In the first case (i.e., the source component contains an update reference), the data that designates the source component's location is modified in the reference to designate the new location of that source component. In the second case (i.e., when another component contains the update reference that refers to the source component), the update reference is changed to identify the new location of the source component.--

9. Please replace the paragraph beginning at the bottom of page 15 and continuing at the top of page16 with the following:

B9

--The previous discussion describes references that can be associated with (i.e., attributes of) other components in the component hierarchy. Many different data structuring techniques can

be used to associate references and components. For example, a mate reference association may be created by storing pointers as part of a common ancestor component's data structure. The pointers can identify the mated components by pointing to their data structures. Alternatively, a mate reference (or other reference type) can be directly inserted using a referencing component. The referencing component may be positioned as the lowest common ancestor of the components referenced. The branches that extend below the mate **reference** node reach the components that are referenced. As the hierarchical data structure is walked, the software determines if a node is a component (i.e., a part or a subassembly), or if the component is a reference component **(e.g., a mate or other reference component)**, and handles that component accordingly. For example, with respect to mate references, earlier disclosure referred to locating common ancestors of mates. In an implementation using reference components, **a** ~~**the**~~ reference component**s** may be created in the restructured hierarchy as a "surrogate" common ancestor**s**. The location of this "surrogate" common ancestor may be determined by locating a "true" common ancestor of the restructured components**,** ~~**and**~~ then ~~**appending**~~ the "surrogate" common ancestor (which will be used to store the mating reference) **can be appended** as a descendent of that "true" common ancestor.--